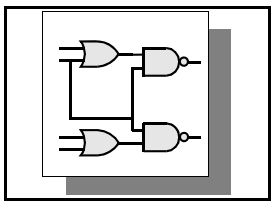
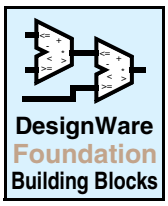




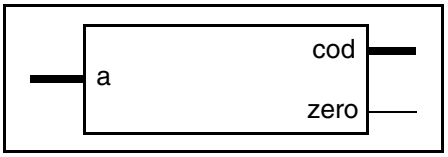
DW_pricod

Priority Coder

Version, STAR, and myDesignWare Subscriptions: IP Directory

Features and Benefits

- Parameterized word length
- Inferable using a function call



Description

The cod output of DW_pricod is a coded one-hot value of the a input vector with a 1 at the most significant (left-most) non-zero bit position of a. All lower order bits (to the right) from the first occurrence of a 1 on the a input port are "don't care." The zero output indicates whether all bits of input a are 0. If a 1 is not found and only 0s are present, the resulting value of cod is all 0s and the value of zero is 1.

Table 1-1 Pin Description

Pin Name	Width	Direction	Function
a	*a_width*	Input	Input vector
cod	*a_width*	Output	One-hot coded value of a.
zero	1	Output	All-zero flag (= 1 if all bits of a are 0)

Table 1-2 Parameter Description

Parameter	Values	Description
a_width	≥1	Vector width of input a

Table 1-3 Synthesis Implementations

Implementation Name	Function	License Feature Required
rtl	Synthesis model	DesignWare
cla	Synthesis model	DesignWare

Table 1-4 Simulation Models

Model	Function
DW01.DW_PRICOD_CFG_SIM	Design unit name for VHDL simulation
dw/dw01/src/DW_pricod_sim.vhd	VHDL simulation model source code
dw/sim_ver/DW_pricod.v	Verilog simulation model source code

Table 1-5 Truth Table (a_width = 8, cod width = 8)

a(7:0)								cod(7:0)								zero
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	1	X	0	0	0	0	0	0	1	0	0
0	0	0	0	0	1	X	X	0	0	0	0	0	1	0	0	0
0	0	0	0	1	X	X	X	0	0	0	0	1	0	0	0	0
0	0	0	1	X	X	X	X	0	0	0	1	0	0	0	0	0
0	0	1	X	X	X	X	X	0	0	1	0	0	0	0	0	0
0	1	X	X	X	X	X	X	0	1	0	0	0	0	0	0	0
1	X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	0

Related Topics

- Math – Arithmetic Overview
- DesignWare Building Block IP User Guide

HDL Usage Through Function Inferencing - VHDL

```
library IEEE,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DW_Foundation.all;
-- If using numeric types from std_logic_arith package,
-- comment the preceding line and uncomment the following line:
-- use DWARE.DW_Foundation_arith.all;

entity DW_pricod_func is

  generic (
    func_a_width : POSITIVE := 8);

  port (
    func_a   : in  std_logic_vector(func_a_width-1 downto 0);
    cod_func : out std_logic_vector(func_a_width-1 downto 0));

end DW_pricod_func;


architecture func of DW_pricod_func is

begin

  -- Function inference of DW_pricod
  cod_func <= DWF_pricod (func_a);

end func;


-- pragma translate_off
configuration DW_pricod_func_cfg_func of DW_pricod_func is
  for func
  end for;
end DW_pricod_func_cfg_func;
-- pragma translate_on
```

HDL Usage Through Function Inferencing - Verilog

```
module DW_pricod_func (func_a, cod_func);

  parameter func_a_width = 8;

  // Passes the width to DW_pricod_function
  parameter a_width = func_a_width;

`include "DW_pricod_function.inc"

  input  [func_a_width-1 : 0] func_a;
  output [func_a_width-1 : 0] cod_func;

  // Function inference of DW_pricod
  assign cod_func = DWF_pricod (func_a);

endmodule
```

HDL Usage Through Component Instantiation - VHDL

```
library IEEE,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DW_Foundation_comp.all;
-- If using numeric types from std_logic_arith package,
-- comment the preceding line and uncomment the following line:
-- use DWARE.DW_Foundation_comp_arith.all;


entity DW_pricod_inst is

  generic (
    inst_a_width : POSITIVE := 8);

  port (
    inst_a    : in  std_logic_vector(inst_a_width-1 downto 0);
    cod_inst  : out std_logic_vector(inst_a_width-1 downto 0);
    zero_inst : out std_logic);

end DW_pricod_inst;


architecture inst of DW_pricod_inst is

begin

  -- Instance of DW_pricod
  U1 : DW_pricod
    generic map (a_width => inst_a_width)
    port map (a => inst_a, cod => cod_inst, zero => zero_inst);

end inst;


-- pragma translate_off
configuration DW_pricod_inst_cfg_inst of DW_pricod_inst is
  for inst
  end for;
end DW_pricod_inst_cfg_inst;
-- pragma translate_on
```

HDL Usage Through Component Instantiation - Verilog

```
module DW_pricod_inst (inst_a, cod_inst, zero_inst);

  parameter inst_a_width = 8;


  input  [inst_a_width-1 : 0] inst_a;
  output [inst_a_width-1 : 0] cod_inst;
  output zero_inst;

  // Instance of DW_pricod
  DW_pricod #(inst_a_width)
    U1 ( .a(inst_a), .cod(cod_inst), .zero(zero_inst) );

endmodule
```

Copyright Notice and Proprietary Information

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html.

All other product or company names may be trademarks of their respective owners.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
www.synopsys.com